网络IO

2020年4月20日 23:26

-
- 网络IO

1. 基本方式
   a. 阻塞与否取决于线程等待调用结果时的状态，如是否被挂起或等待
   b. 非阻塞一般会提前返回一个提示，表示需要等一会
   c. 异步与同步强调是否能并行做其他工作
2. 常用的IO通信模型
   a. BIO同步阻塞Blocking IO
      i. 如new ServerSocket(端口号).accept().getIn/OutputStream()
      ii. java.net的实现

```
ServerSocket serverSocket = new ServerSocket(8888);
Socket socket = serverSocket.accept();
InputStream in = socket.getInputStream();
OutputStream out = socket.getOutputStream();
可用setSoTimeout()方法设置超时时间和多线程改进
```

   b. NIO同步非阻塞Non-blocking IO
   c. IO多路复用IO Multiplexing
      i. 适合高并发场景（低并发时效率并不高）
      ii. 主流的实现技术：
         1) Select/poll轮询所有并发socket
         2) epoll使用callback机制，无连接限制，减少内存数据拷贝
         3) callback回调：监听到信道有新数据时调用
      iii. java1.4的NIO的实现
         1) 为所有的原始类型提供 (Buffer) 缓存支持。
            a) 是一块连续的内存块。
            b) 是 NewIO 数据读或写的中转地。
         2) 字符集编码解码解决方案。
         3) Channel ： 一个新的原始 I/O 抽象。
            a) 数据的源头或者数据的目的地
            b) 用于向 buffer 提供数据或者读取 buffer 数据 ,buffer 对象的唯一接口
            c) 异步 I/O 支持
            d) Selector：一组socketChannel的代理，负责事件订阅和Channel管理
         4) 支持锁和内存映射文件的文件访问接口。
         5) 提供多路 (non-bloking) 非阻塞式的高伸缩性网络 I/O 。
      iv. 优点：不用多线程，速度和性能较高，单端口可绑定多个协议
   d. 信号驱动式IO
   e. AIO异步Asynchronous IO
      i. java1.7的NIO2.0增加了异步IO实现
         1) 采用“订阅-通知”模式，应用程序直接向操作系统注册IO监听，然后异步运行其他程序，待操作系统准备好数据再主动通知程序
         2) 此时channel不需要selector了

3. Netty：一个NIO C/S框架，能够快速、简单的开发协议服务器和客户端等网络应用。能很大程度上简单化、流水线化开发网络应用，例如TCP/UDP socket服务器
    a. 基于JAVA NIO 多路复用IO技术，是NIO的高层封装，性能高，使用方便简洁
    b. 关键：线程机制、ByteBuf缓存机制、高度抽象的专用于网络的Channel
    c. 优势：
        i. 支持多种网络协议和多种数据信息格式封装
        ii. 支持多种I/O模型
        iii. 对复杂的网络操作进行了高层抽象，使用更加简单。
        iv. 弥补了java i/o的一些不足和Bug
    d. 应用场景：
        i. RPC的网络通信和传输
        ii. 特殊网络服务器的通信接口（各种协议如HTTP协议）
        iii. 特殊网络要求（如长连接等）